3D Surface Modeling from Curves

[a] *Laboratoire de Vision et Systèmes Numériques*
Département de génie électrique et de génie informatique
Université Laval
Sainte-Foy (Québec), Canada G1K 7P4

Dragan Tubić [a] Patrick Hébert [a] Denis Laurendeau [a]

Abstract

Traditional approaches for surface reconstruction from range data require that the input data be either range images or unorganized sets of points. With these methods, range data acquired along curvilinear patterns cannot be used for surface reconstruction unless constraints are imposed on the shape of the patterns or on sensor displacement. This paper presents a novel approach for reconstructing a surface from a set of arbitrary, unorganized and intersecting curves. A strategy for updating the reconstructed surface during data acquisition is described as well. Curves are accumulated in a volumetric structure in which a vector field is built and updated. The information that is needed for efficient curve registration is also directly available in the vector field. The proposed modeling approach combines surface reconstruction and curve registration into a unified procedure. The algorithm implementing the approach is of linear complexity with respect to the number of input curves and makes it suitable for interactive modeling. Simulated data based on a set of six curvilinear patterns as well as data acquired with a range sensor are used to illustrate the various steps of the algorithm.

Keywords: 3D modeling, surface reconstruction, 3D curves, geometric fusion, curve registration, volumetric representation

1 Introduction

Three-dimensional modeling from 3D (range) data is the process of building a 3D surface model of a measured object. Depending on the type of the sensor, the range data can be a set of unorganized points, surface curves or range images (surface patches). In all cases however, range data acquired from a single viewpoint is not sufficient to build a complete model; thus the 3D sensor or the object have to be moved in order to observe the whole surface. This creates *redundancy* in acquired data with respect to the resolution of the final model since, in general, it is practically not possible to observe the

whole surface without observing some regions two or more times. When the range data is a set of surface patches, building a model consists in removing this redundant data and reparameterizing the surface. This process is usually referred to as *geometric fusion*. Building a model of a given resolution from curves or unorganized sets of points also requires removing redundant data as well but in addition, *surface reconstruction* is necessary since curves and unorganized points do not represent a surface. Although the redundancy of range data might appear as a setback, it is however very useful in solving another problem, registration of multiple views.

Alignment, or *registration* of range data is required if the sensor position and orientation are not perfectly known for each viewpoint. That information is usually obtained by using an external positioning device but some sensors can also estimate their position using features present in the observed scene [7]. Since the estimate of sensor position is always inaccurate to some degree, the range data is not perfectly aligned once it is transformed into a common reference frame. This sensor position error is referred to as the *registration error*. Note that in 3D modeling it is assumed that an *approximate* position of the sensor is known since the commonly used algorithms cannot handle arbitrary initial positions. For this reason, registration is more correctly referred to as *pose refinement* in the context of 3D modeling. To align two views it is sufficient to identify at least three common points on each view and compute a rigid transformation that aligns them. Selecting common points is referred to as *matching* and is usually based on the ICP (Iterated Closest Points) algorithm [2] while numerous variants and specific implementations exist. In this paper, the process of three-dimensional modeling integrates geometric fusion, surface reconstruction and registration.

A very common type of range sensor acquires curvilinear measurements (surface profiles) on the surface of an object. Based on optical triangulation, these sensors use a focused laser pattern projected on the surface and can provide dense profiles robustly in a very short time frame. Although various laser patterns are available, methods for surface reconstruction from arbitrary curves still need to be developed. Currently, it is often assumed that profiles are gathered at regular intervals along a well defined path allowing neighbouring profiles to be grouped into a surface. The relative pose of each profile is assumed to be accurate. For hand-held sensors that collect profiles, it is not possible to scan a surface along an exactly predefined scanning path. Thus, some methods have been proposed to build local surface patches from a rigid series of profiles with known position but with only nearly regular scanning paths [3,9,11,15]. There are still two problems with these methods. First, the only type of surface curves that can be used are surface profiles. Second, although error in sensor positioning cannot be avoided during acquisition, it is not possible to refine the pose for a single profile. This results in a loss of quality of the model. The current alternative for low cost hand-held acquisition

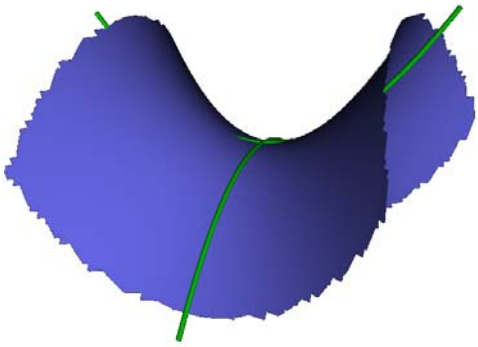


Fig. 1. Example of surface reconstruction from two curves.

is to collect range images [6,14] at the expense of losing robustness of laser sensors. In this paper, an approach is proposed for surface reconstruction of free-form objects from arbitrary curves as well as for the correction of the pose of each individual curve.

Surface curves contain more information about the measured surface than an unorganized set of points, namely tangents that can be exploited to improve the quality of the reconstructed surface. A local estimate of the surface can be obtained using tangents of intersecting curves. Later in this paper, it is explained that for each point on the reconstructed surface, its surface normal is assessed from the tangents at the closest points on the neighbouring curves. Figure 1 illustrates the simplest case where a surface is reconstructed in the neighbourhood of two intersecting curves. Since the surface is approximated, the error of the reconstructed surface increases as the distance from the intersection increases. However, one can observe that the reconstructed surface still faithfully follows the shape of the original surface. This cannot be accomplished by reducing curves to unorganized sets of points, i.e. by fitting a plane in the neighbourhood of a point (see also Figure 3). This improvement is particularly important in the regions of uneven sampling density such as the example in Figure 1.

It is also shown that surface reconstruction can be performed efficiently using a volumetric structure where the surface is incrementally recovered as new surface curves are scanned. Volumetric structures have been exploited for surface reconstruction from range images or unordered sets of points [4,9,13,15,17] to reduce computational complexity and provide incremental reconstruction. These approaches use an implicit representation of the surface e.g. a scalar signed distance field computed on a volumetric grid where the reconstructed surface corresponds to the zero crossings of the field. *The first contribution* of this paper is to describe how such a volumetric representation can be built directly from a set of measured curves using their tangents without any inter-

mediate surface representation. Furthermore, the computational complexity with respect to the number of curves is linear and the reconstruction is incremental and order independent; thus no constraints are imposed on sensor displacement. Any light pattern can be used to measure curves as long as the curves intersect.

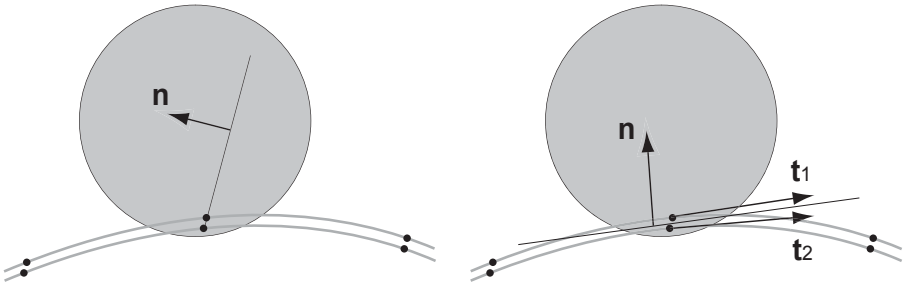
If one not only encodes the signed distance field in the volumetric structure but also includes the direction towards the nearest zero crossing in each voxel, then matching (closest point) for a point can be obtained directly from the nearest voxel. We have developed this idea to provide near real-time registration of range images [18]. *The second main contribution* in this paper is to still maintain the same linear complexity registration for surface curves by registering them with the reconstructed surface. Another important aspect of our approach is a novel definition of distance measure that improves robustness of registration in the presence of noise.

Due to the shadow effects, some regions of the object are not measured during the acquisition of range data, resulting in holes in the final model. The object then has to be rescanned to complete the model in these regions. It is thus advantageous to provide a partially reconstructed model in real-time to guide the operator for the selection of the next best view that allows completion of the model with minimal acquisition time. The linear computational complexity of all modeling steps in the approach makes on-line surface reconstruction and registration feasible.

The paper is organized as follows. The next section presents a summary of existing methods for surface reconstruction and registration. Sections 3.1 and 3.2 describe the principle of surface reconstruction from curves, followed by section 3.3 which explains a modification required to allow incremental reconstruction. A novel definition of distance aimed at improving robustness of registration is presented in section 3.4. Section 3.5 exposes curve registration. The results presented in section 4 illustrate each aspect of the approach using simulated and real data.

2 Related Work

Reconstruction methods from range data (geometric fusion of multiple views) can be divided into two groups based on the representation of a surface: surface based approaches [16,19] and volumetric approaches [4,9,10,13,15]. In this paper we concentrate on the latter group. Volumetric approaches to surface reconstruction and fusion are based on implicit surface representation as a signed distance field. The surface itself is the zero-set of the scalar field. In the case of range images, this scalar field is used to merge multiple views by



(a) (b)

Fig. 2. Estimating local orientation of the surface from two different viewpoints represented as two curves. a) In low sample density regions, small positioning errors can lead to a wrong orientation estimate if a plane is fitted in a neighbourhood of points (shaded circle). b) Using the normals or tangents computed from connectivity of input data solves this problem. In this example the normal $\mathbf{n}$ is obtained as "the most perpendicular" vector to tangents $\mathbf{t}_1$ and $\mathbf{t}_2$. By doing so, the effect of the translation that caused a wrong orientation estimate in (a) is canceled.

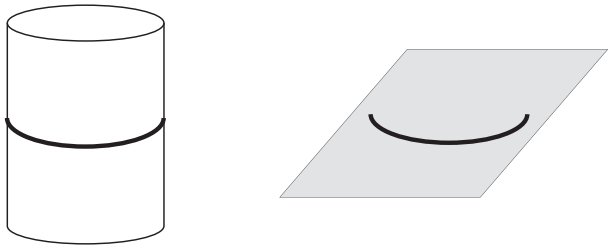


Fig. 3. Considering all 3D data as unorganized sets of points might lead to wrong surface reconstruction. In this example, the surface reconstructed from a measured curve (left) is a plane perpendicular to the real surface (right).

simply doing a summation of the fields for each view. The same scalar field can also be used to reconstruct the surface from unorganized sets of points [10,15]. In this case, the distance is defined as the distance to a plane fitted in a neighbourhood of the point where the field is being computed. The final surface is extracted using the Marching Cubes algorithm [12]. Besides geometric fusion, Masuda [13] merges registration and reconstruction by aligning signed distance fields. This approach only applies if the input range data is a set of surface patches.

Most surface reconstruction algorithms using range data accept either range images [4,13,16] or unorganized sets of points [1,10,15] as input. There are several problems related to surface reconstruction by considering curves as unorganized sets of points. First, a local estimate of the surface orientation has to be inferred from relative positions of the measured 3D points, as illustrated in Figure 2a. This can be accomplished for example by fitting a plane in a neighbourhood of points. However, if the 3D data consists of multiple views, as in Figure 2, a very small positioning error can cause a very inaccurate estimate of surface orientation. Another way to proceed and to circumvent this

problem is to use additional information that is contained in the connectivity of points in curves and surface patches. For example, on a triangulated surface, the immediate neighbours for each point are known and can be used to estimate the surface normal at each point. In this case the surface orientation is estimated as the average of normals in a neighbourhood, see Figure 2b. By doing so, the effect of the registration error is much smaller than in the previous case. Similarly, curve tangents can be used to improve reconstruction and reduce the effect of registration errors as explained later.

Another reason to use connectivity information in 3D data (tangents or normals) is the ambiguity that leads to completely wrong surface reconstruction. One example of such an ambiguity is shown in Figure 3 where a single curve is measured on a cylindrical object. By considering this curve as an unorganized set of points, the reconstructed surface will inevitably be a plane that contains the curve and will be perpendicular to the real surface. The reason for this is that curves are dense sets of points but only along one dimension, i.e the distance between curves is not necessarily the same as the distance between neighbouring points within a single curve. This means that, for example, the surface shown in Figure 1 could not be reconstructed using surface reconstruction algorithms from unorganized points since they generally assume a relatively uniform distribution of points over the surface. The method proposed in this paper for surface reconstruction from curves eliminates this problem by using curve tangents to obtain a local estimate of the surface.

With the exception of [9], none of the proposed surface reconstruction methods is applicable to the reconstruction from surface curves. Nevertheless, the method proposed in [9] assumes that the input data is a set of surface profiles which can be locally triangulated to produce a surface. Once the profiles are grouped into a surface, it is considered as a surface patch so that a volumetric algorithm can be used to reconstruct the surface. The triangulation step has a negative impact on several aspects of modeling: the registration of individual profiles cannot be performed, range data has to be acquired in regular sequences and the model cannot be constructed incrementally by integrating a single profile at a time. A post processing algorithm [3] for Hilton's method has been proposed to improve the alignment of profiles as well as to optimize reconstructed surface. However, the algorithm [3] cannot register individual curves and a specialized algorithm for this purpose is required.

Very little work has been reported on the subject of surface curve registration [8]. Although the stability of curve registration may have contributed to this, the main reason is the computational complexity of the registration process. The simultaneous registration based on the ICP (Iterated Closest Points) algorithm [2] is of complexity $O(N^2)$ with respect to the number of range images. If the same algorithm is applied for registration of surface curves, the complexity quickly becomes untractable since the number of curves required for

complete reconstruction of an object is at least an order of magnitude larger than the equivalent number of range images and can easily reach several thousand curves.

A different approach for registration has been proposed in [17] where matching information is explicitly encoded in a volumetric grid in addition to the distance field. Such a vector representation reduces computational complexity of registration and makes it linear with respect to the number of range images and the number of measured 3D points. Besides surface reconstruction, this paper extends this approach to the registration of surface curves where the computational complexity is linear with respect to the number of curves.

3 Modeling using Range Curves

As reported in an earlier paper [17], the reconstructed surface is represented implicitly as a vector field. Such a *volumetric* representation contains both the reconstructed surface and its corresponding matching information in the form of direction and distance towards the reconstructed surface. The surface is represented as the zero-crossing of the signed norm of the field while the distance and direction are represented by the field itself. Encoding both distance and direction towards the surface allows one to find approximate closest points (required for registration) with linear complexity. Another advantage of this volumetric representation is that the reconstructed surface can be updated incrementally. In the following, we describe how an implicit representation of the surface can be obtained from a set of non-parallel, intersecting surface curves.

3.1 Reconstruction from two intersecting curves

The main idea behind our approach is to perform reconstruction by *approximating* the surface in the neighbourhood of the intersection points of surface curves. The reconstruction from two intersecting curves is first explained in order to illustrate the principle on a simple case.

The reconstruction is based on a fundamental property of differential surfaces stating that all surface curves passing through some point have their tangents located in a plane tangent to the surface at the same point [5]. The most important consequence of this property for our application is that the tangent plane to a surface at a given point can be computed using the vector product of the tangent to each curve at this point.

In theory, planes at the intersection points of surface curves could be used to roughly approximate a surface. However, a faithful reconstruction would require too many intersections to be practical. In the presented approach, a limited number of intersections is used and a faithful surface representation is built using tangent planes at these intersections and at points in their neighbourhoods.

The reconstructed surface $\hat{S}$ is implicitly represented as a vector field $\mathbf{f} : R^3 \rightarrow R^3$ where $\mathbf{f}(\mathbf{p})$ represents the direction and the distance to the closest point $\mathbf{p}_c$ on the surface $\hat{S}$:

$$\mathbf{p} + \mathbf{f}(\mathbf{p}) = \mathbf{p}_c \in \hat{S}, \tag{1}$$

such that

$$\mathbf{p}_c = \underset{\mathbf{q} \in \hat{S}}{\operatorname{argmin}}\, d(\mathbf{p}, \mathbf{q}), \tag{2}$$

where d denotes a distance measure. In practice, the field $\mathbf{f}(\mathbf{p})$ is computed at points on a regular lattice (volumetric grid, see Figure 4) in the vicinity of a surface, usually referred to as its envelope. The envelope encloses lattice points which are located at a distance to the surface smaller than a predefined positive value ϵ. An example of such an envelope for a curve is depicted in Figure 4.

To compute the implicit representation of the surface $\mathbf{f}(\mathbf{p})$ we start with two surface curves $\alpha_1 : U_1 \subset R \rightarrow R^3$ and $\alpha_2 : U_2 \subset R \rightarrow R^3$ intersecting at a single point $\mathbf{p}_i = \alpha_1(U_1) \cap \alpha_2(U_2)$ (see Figure 6). In the following it is assumed that the surface is differentiable and that curves are parameterized by arc length. Now, for some lattice point (voxel) $\mathbf{p}$, an approximation of the normal $\mathbf{n}$ to the tangent plane at the closest point of the surface is approximated as the vector product of the tangents $\mathbf{t}_1$ and $\mathbf{t}_2$ at points $\mathbf{p}_1 \in \alpha_1$ and $\mathbf{p}_2 \in \alpha_2$ closest to $\mathbf{p}$, i.e.

$$\mathbf{n} = \mathbf{t}_1 \times \mathbf{t}_2, \tag{3}$$

where

$$\begin{aligned}
\mathbf{p}_1 &= \underset{\mathbf{q} \in \alpha_1}{\operatorname{argmin}}\, d(\mathbf{p}, \mathbf{q}) = \alpha_1(u_1), u_1 \in U_1, \\
\mathbf{p}_2 &= \underset{\mathbf{q} \in \alpha_2}{\operatorname{argmin}}\, d(\mathbf{p}, \mathbf{q}) = \alpha_2(u_2), u_2 \in U_2, \\
\mathbf{t}_1 &= \alpha_1'(u_1), \\
\mathbf{t}_2 &= \alpha_2'(u_2)
\end{aligned} \tag{4}$$

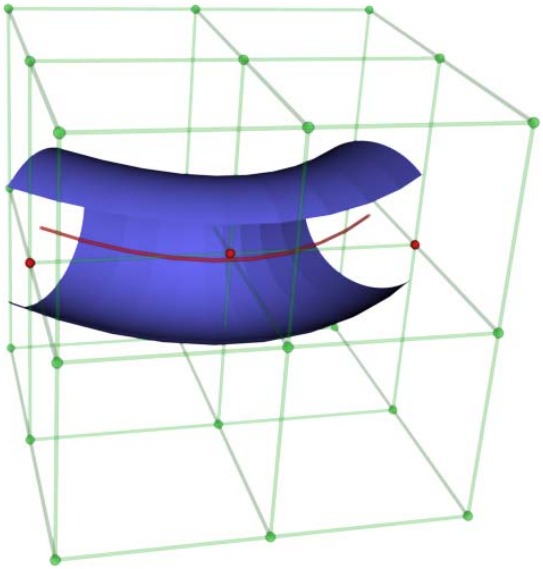
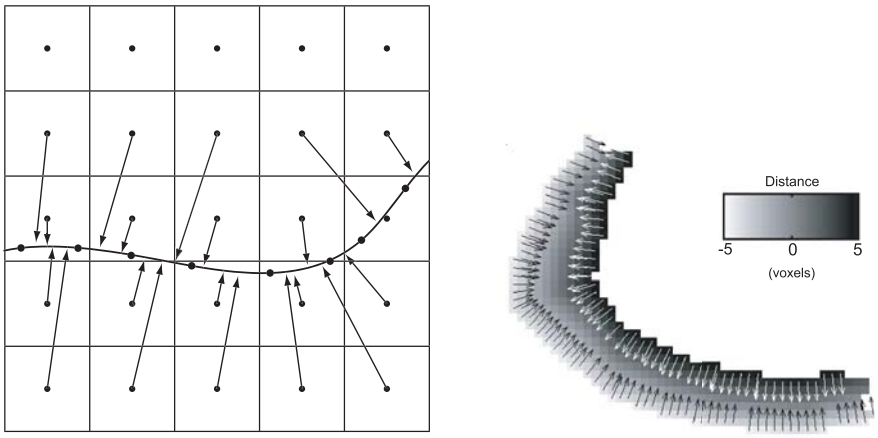


Fig. 4. Example of a volumetric envelope (blue) for a curve. The volumetric envelope contains all points closer to the curve than a predefined constant distance $\epsilon > 0$. A cylindrical *iso-surface* whose points are located at a distance ϵ from the curve delimits the envelope. Field $\mathbf{f}(\mathbf{p})$ is computed only at grid points (red dots) located inside the envelope.



(a) (b)

Fig. 5. Example of the vector field computed on a regular grid. a) The values of the field represent the direction and the distance towards the closest point on the surface as defined in Eq. 1. b) Norm of the vector field encoded as gray-levels with associated directions towards the surface.

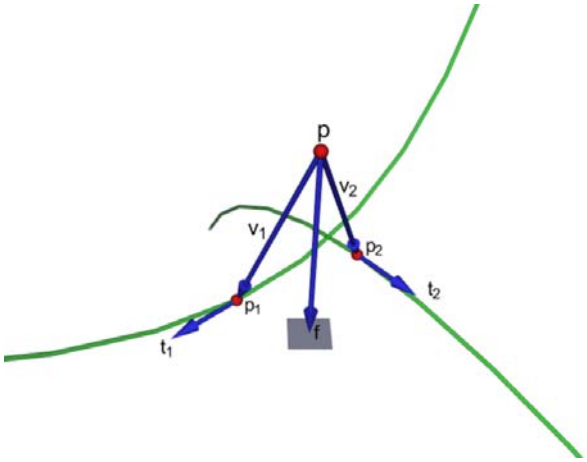


Fig. 6. Reconstruction from two curves. The normal to the tangent plane (shaded in gray) is obtained as the cross product of the tangents $\mathbf{t}_1$ and $\mathbf{t}_2$ at the closest points $\mathbf{p}_1$ and $\mathbf{p}_2$ of the two curves α_1 and α_2. The distance d to the tangent plane is the average of the projections of $\mathbf{v}_1$ and $\mathbf{v}_2$ on the normal $\mathbf{n}$. The value of the field $\mathbf{f}$ at point $\mathbf{p}$ is $d \cdot \mathbf{n}$.

and d is a distance measure. As discussed in section 3.4, the choice of d in Eq. 4 plays an important role in the registration process, but for now, it can be assumed that the distance d is the Euclidean distance.

The normal $\mathbf{n}$ computed in Eq 3, is taken as the direction towards the closest point on the surface. The average of the projections of vectors $\mathbf{v}_1 = \mathbf{p} - \mathbf{p}_1$ and $\mathbf{v}_2 = \mathbf{p} - \mathbf{p}_2$ on the normal $\mathbf{n}$ is taken as the distance to the surface, i.e.

$$d(\mathbf{p}, S) = \frac{1}{2}\langle \mathbf{v}_1 + \mathbf{v}_2, \mathbf{n} \rangle. \tag{5}$$

And the value of field $\mathbf{f}$ at $\mathbf{p}$ is computed as:

$$\mathbf{f}(\mathbf{p}) = d(\mathbf{p}, S) \cdot \mathbf{n}. \tag{6}$$

An example of the vector field is shown in Figure 5. The reconstructed surface can be extracted from the vector field $\mathbf{f}(\mathbf{p})$ using the Marching Cubes algorithm. The Marching Cubes algorithm uses a scalar field that is obtained by computing the signed norm of $\mathbf{f}(\mathbf{p})$. This norm is a scalar field whose sign is computed as the sign of the scalar product between $\mathbf{f}(\mathbf{p})$ and the direction of the sensor. An example of the reconstruction from two curves is shown in Figure 1.

It should be noted that as long as the two curves are not straight lines, there are no planar regions on the reconstructed surface: the tangent planes are used to approximate the surface but each plane represents only a single point on the surface.

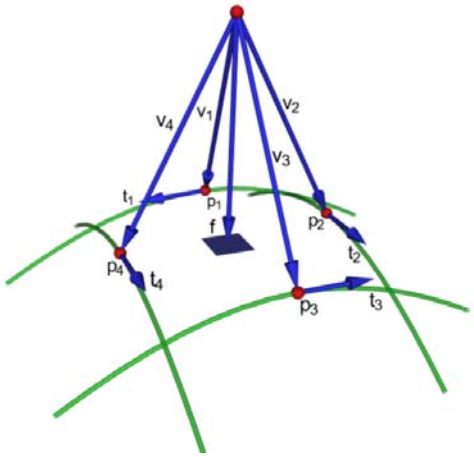


Fig. 7. Reconstruction from multiple curves. The normal is obtained as a least-squares estimate of the "most perpendicular" vector to the tangents $\mathbf{t}_i$, $i = 1, 2, 3, 4$ at the closest points $\mathbf{p}_i$ of the four curves. Distance d to the tangent plane is the average norm of the projections of vectors $\mathbf{v}_i$ on the normal $\mathbf{n}$. The value of the field $\mathbf{f}$ at point $\mathbf{p}$ is $d \cdot \mathbf{n}$.

3.2 Reconstruction from multiple curves

It is straightforward to extend the two-curve reconstruction approach described in section 3.1 to the case of multiple curves. As illustrated in Figure 7 a voxel can be located in the vicinity of more than two curves and the tangent plane has to be estimated using all of them. The approximation of the surface can be obtained as a least-squares estimate of the normal using tangents at the closest points of *all* nearby curves. This estimate corresponds to the "most perpendicular" vector to a set of tangents.

More formally, let $\alpha_1, \ldots, \alpha_N$ be N surface curves passing within some predefined distance $\epsilon > 0$ from a point (voxel) $\mathbf{p}$ and let $\mathbf{t}_1, \ldots, \mathbf{t}_N$ be their respective tangents at the closest points to the point $\mathbf{p}$. Then the normal on the surface is obtained as the vector $\mathbf{n} = [n_x, n_y, n_z]^T$ that minimizes the following expression

$$\xi = \sum_{i=1}^{N} \langle \mathbf{t}_i, \mathbf{n} \rangle^2 \tag{7}$$

Taking the derivatives of ξ with respect to n_x, n_y and n_z and setting them equal to zero defines the following system of equations:

$$\frac{1}{N} \sum_{i=1}^{N} \mathbf{t}_i \mathbf{t}_i^T \mathbf{n} = C\mathbf{n} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T . \tag{8}$$

The solution for $\mathbf{n}$ is the eigenvector associated with the smallest eigenvalue of C which is the covariance matrix of the tangents.

The distance towards the surface is obtained as the average value of the projected distance vectors on the estimated normal, i.e

$$d(\mathbf{p}, \hat{S}) = \frac{1}{N} \sum_{i=1}^{N} \langle \mathbf{p}_i - \mathbf{p}, \mathbf{n} \rangle = \langle \tilde{\mathbf{v}}, \mathbf{n} \rangle \tag{9}$$

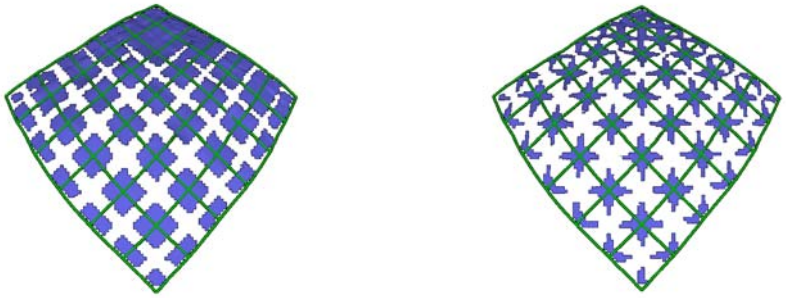
where $\mathbf{p}_i$ is the closest point to $\mathbf{p}$ on curve α_i. Finally, the value of the field at point $\mathbf{p}$ is:

$$\mathbf{f}(\mathbf{p}) = d(\mathbf{p}, S) \cdot \mathbf{n}. \tag{10}$$

In order to estimate the tangent plane to a surface at some point $\mathbf{p}$, at least two non-parallel tangents are needed to compute the matrix C at $\mathbf{p}$ since a single tangent does not define a plane. This condition can be verified by analyzing the eigenvalues of matrix C: if two eigenvalues are zero, then only one tangent (or two or more parallel tangents) exists and the estimated normal at that point is not used. This implies that the tangent plane cannot be estimated from parallel curves, which makes the estimate less sensitive to registration errors (relative position of curves).

At the intersection points of noiseless and accurately positioned curves, all tangents are coplanar, hence one eigenvalue is always equal to zero and the tangents span a plane. Since, in our case, the surface is *approximated* in the neighbourhood of the intersection points, all three eigenvalues are generally larger than zero. Also, if the tangents are estimated from noisy data, the three eigenvalues may have similar values in which case the tangents span a cube and the estimate of the tangent plane is meaningless. To make sure that the estimated tangent plane is valid, an additional verification is made on the eigenvalues. Let e_1, e_2 and e_3 be the three eigenvalues such that $e_1 < e_2 < e_3$. Since matrix C is normalized, the sum of the eigenvalues is always equal to one i.e. $e_1 + e_2 + e_3 = 1$ and fixed thresholds can be applied to test eigenvalues. It was confirmed empirically, that imposing the constraints $e_2 > 0.05$ and $e_1 < 0.5e_2$ is enough to validate the estimated tangent plane. Imposing these constraints simply means that the tangents used to compute C must be approximately coplanar.

Since the size of the envelope determines how many curves influence the matrix C at each voxel, its role is important in the reconstruction process: If the envelope chosen is too small with respect to the density of the curves, the reconstructed surface will either contain holes or it will appear as a set of patches located around intersection points. Two examples of a surface re-



(a) (b)

Fig. 8. Surface reconstruction using a small envelope size with respect to the density of range data. The shape of the reconstructed surface patches depends on the weighting function. a) Example of reconstruction using unweighted matrices C_i. b) Reconstruction using the weighting function defined in Eq. 11.

constructed with too small an envelope are shown in Figure 8. As illustrated in Figure 9, increasing the size of the envelope can solve this problem. Nevertheless, increasing the size of the envelope also increases the execution time of the algorithm since the field is computed for a larger number of points. More importantly, by increasing the size of the envelope, the number of tangents used for estimating the tangent plane increases, as well as the distance to the curves whose tangents are used. The consequence is a loss of details on the reconstructed surface since the least-squares estimation of the tangent plane acts as a low-pass filter. Having these constraints in mind, choosing the size of the envelope is thus a compromise. It is preferable to solve the problem of sparse data by sampling the surface more densely. Since our approach aims at interactive modeling where the reconstructed surface becomes available immediately after each curve has been acquired, it is easy to spot and resample the low-density regions during the acquisition of range data.

A single curve influences the field only at points located inside its envelope and its influence drops to zero outside this envelope. Thus, the field computed from Eq. 10 is discontinuous over the edges of the envelope as well as the reconstructed surface. An example of such a reconstructed surface is shown in Figure 9. The solution to this problem is to weight the tangents using a continuous function of distance that drops to zero at the edge of the envelope. Any decreasing, monotonic function can be used for this purpose. In our experiments the following function proved to be useful:

$$\omega(d) = e^{-d^2/\sigma^2}. \tag{11}$$

The value of σ is chosen equal to $1/2\epsilon$ to make sure that the value of ω drops close to zero outside the envelope. The weighting function also influences the shape of the reconstructed surface since it also acts as a low-pass filter

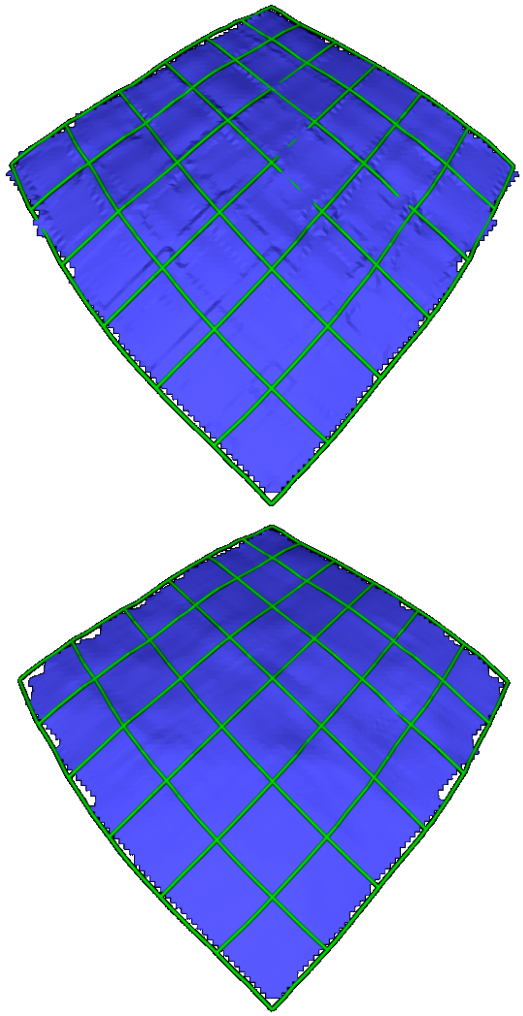


Fig. 9. Reconstruction using an envelope size larger than the minimum distance between curves. If unweighted tangents are used for computing the field, the resulting field is discontinuous and so will be the reconstructed surface. If the surface does not change abruptly, the effect of discontinuity is relatively small, giving a "patchy" appearance to the reconstructed surface, as shown in the image on the top. The image on the bottom shows the same surface reconstructed using the weighting function defined in Eq. 11.

giving more importance to closer curves. As illustrated in Figure 8, the shape of the reconstructed surface patches around intersection points is influenced by the weighting function. This results from the constraints imposed on the eigenvalues of matrix C.

The tangents can also be weighted in such a way to reduce influence of less-confident data, by taking into account some uncertainty measure τ, based for example on the angle between the curve normal and the incident measurement

ray. After weighting with τ and ω, matrix C defined in Eq. 8 becomes

$$C = \frac{1}{\sum_{i=1}^{N} \tau_i \omega_i} \sum_{i=1}^{N} \tau_i \omega_i \mathbf{t}_i \mathbf{t}_i^T. \tag{12}$$

3.3 Incremental reconstruction

For the reconstruction approach described above, it was assumed that all the data has been collected prior to reconstructing the surface. If the reconstruction has to be performed online, then the field needs to be updated incrementally by integrating a single measured curve at a time. However, the least-squares estimate of the surface normal and, consequently, the vector field, cannot be computed incrementally. On the other hand, matrix C computed at each voxel $\mathbf{p}$ is obtained as a sum and can therefore be updated incrementally. Let $C(\mathbf{p})$ be the matrix C for the voxel $\mathbf{p}$. Equation 12 can be rewritten as:

$$C(\mathbf{p}) = \frac{1}{\sum_{i=1}^{N} \tau_i \omega_i} \sum_{i=1}^{N} \tau_i \omega_i C_i(\mathbf{p}), \tag{13}$$

where $C_i(\mathbf{p}) = \mathbf{t}_i \mathbf{t}_i^T$. During reconstruction, a matrix $C(\mathbf{p})$ is associated with each voxel and is updated after each curve α_i has been acquired by summing it with $C_i(\mathbf{p})$. Matrix $C_i(\mathbf{p})$ depends only on the curve α_i and is computed using the tangent $\mathbf{t}_i$ at the point $\mathbf{p}_i$ on the curve that is closest to $\mathbf{p}$. In addition to the matrix C, the sum of distance vectors $\tilde{\mathbf{v}}$ is kept at each voxel as well (see Eq. 9). The value of the field $\mathbf{f}(\mathbf{p})$, defined in equations 9 and 10, is computed only before the application of the surface using the Marching Cubes algorithm for reconstruction or during registration.

3.4 Defining the distance and computing the vector field

So far the curves were considered as being continuous and noiseless. In practice, the measured curves are represented as sets of line segments and corrupted by noise. The level of noise does not have a significant effect on the functionality of the algorithm with respect to the surface reconstruction process other than the quality of the reconstructed surface. The most important effect of noise is on the registration process, specifically on the matching step. As illustrated in Figure 15.a, the points with high noise level tend to attract a large number of correspondences. This slows down the registration process and makes it less accurate. To circumvent this problem, a new definition is adopted for distance d in Eq. 3 to replace the Euclidean distance.

To improve the robustness of matching, the *direction* towards the closest point on the line segment has to be corrected while taking into account two important constraints: i) the distance field has to be continuous, ii) the position of the measured 3D points should not be altered. The solution for this problem is to filter and interpolate tangents over the line segments and to define the distance using the filtered tangents. For the sake of clarity, the distance computation is first illustrated in 2D, the curve being contained in a plane.

A curve is considered as a linear interpolation of measured points $\{\mathbf{p}_1 \dots \mathbf{p}_N\}$ represented as a set of line segments $\alpha = \{l_1, \dots, l_{N-1}\}$ where $l_i = \overline{\mathbf{p}_i \mathbf{p}_{i+1}}$. Except for end-points, the tangent at each measured point $\mathbf{p}_i$ can be computed as follows:

$$\mathbf{t}_i = \frac{1}{2}\left[\frac{\mathbf{p}_{i-1} - \mathbf{p}_i}{\|\mathbf{p}_{i-1} - \mathbf{p}_i\|} + \frac{\mathbf{p}_i - \mathbf{p}_{i+1}}{\|\mathbf{p}_i - \mathbf{p}_{i+1}\|}.\right] \tag{14}$$

The tangents t_i are filtered with a filter Φ whose size is $2N + 1$:

$$\mathbf{t}_i = \sum_{k=i-N}^{i+N} \mathbf{t}_k \Phi(k - i). \tag{15}$$

For the experiments reported herein, a simple low-pass filter,

$$\Phi(i) = \begin{cases} 1/(2N+1) & \text{if } i \in [-N, \dots, N] \ , \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

was chosen.

The distance between some point $\mathbf{p}$ and the curve α is computed with respect to the closest line segment l_c, i.e.

$$\begin{aligned} d(\mathbf{p}, \alpha) &= d(\mathbf{p}, l_c), \\ l_c &= \underset{l \in \alpha}{\operatorname{argmin}}\, d(\mathbf{p}, l), \end{aligned} \tag{17}$$

where d denotes a distance measure. It is therefore sufficient to provide the distance from a point to a line segment. The set of all points $\Pi \subset R^3$ for whom a line segment l_c is the closest, is called a *fundamental cell* associated with the *generator line segment* l_c. For the purpose of surface modeling, the field needs to be calculated only within a relatively small distance $\epsilon > 0$ from the measured curve thus limiting the size of the cell to the set of points which are closer than ϵ, i.e. inside the envelope. If d is the Euclidean distance then the

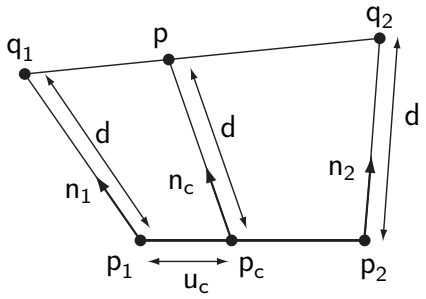


Fig. 10. Computing the distance in the direction of interpolated normals. Distance d between the point $\mathbf{p}$ and the line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$ is defined as the distance between $\mathbf{p}$ and the point $\mathbf{p}_c$ whose normal $\mathbf{n}_c$ passes through $\mathbf{p}$. The distance d is computed by finding the iso-segment $\overline{\mathbf{q}_1\mathbf{q}_2}$ (line segment where each point is located at an equal distance from $\overline{\mathbf{p}_1\mathbf{p}_2}$) that contains the point $\mathbf{p}$.

fundamental cells correspond to the cells of the Voronoi diagram for a set of line segments.

The introduction of a new definition of distance requires the parameterisation of a line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$ as:

$$\mathbf{l}(u) = \mathbf{p}_1 + u(\mathbf{p}_2 - \mathbf{p}_1), \quad 0 \leq u \leq 1. \tag{18}$$

The tangents at the two end-points are then interpolated over the line segment as:

$$\mathbf{t}(u) = \mathbf{t}_1 + u(\mathbf{t}_2 - \mathbf{t}_1), \quad 0 \leq u \leq 1. \tag{19}$$

The normal $\mathbf{n}(u)$ at each point of a line segment is defined as the vector being perpendicular to the tangent and can also be interpolated:

$$\mathbf{n}(u) = \mathbf{n}_1 + u(\mathbf{n}_2 - \mathbf{n}_1), \quad 0 \leq u \leq 1. \tag{20}$$

Finally, the distance d between a point $\mathbf{p}$ and the line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$ is defined as the distance between $\mathbf{p}$ and the point $\mathbf{p}_c$ whose normal $\mathbf{n}_c$ passes through $\mathbf{p}$. This is illustrated in Figure 10. More formally:

$$d(\mathbf{p}, \overline{\mathbf{p}_1\mathbf{p}_2}) = d(\mathbf{p}, \mathbf{l}(u_c)) = e, \qquad 0 \leq u_c \leq 1, \tag{21}$$

such that

$$\mathbf{p} = \mathbf{l}(u_c) + e \cdot \mathbf{n}(u_c). \tag{22}$$

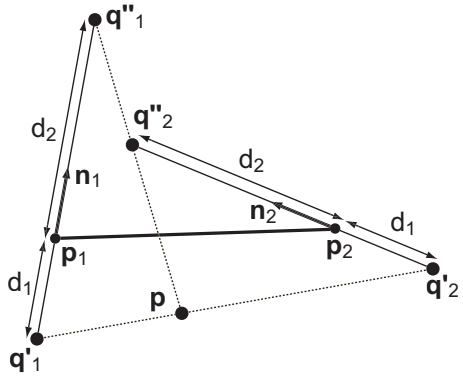


Fig. 11. Illustration of real roots in Eq. 23. Even though the two roots d_1 and d_2 satisfy Eq. 23, only the root with a smaller absolute value represents a valid distance. The other root is larger than the maximal allowed distance within the cell. The point $\mathbf{p}$ is contained in the line segment $\overline{\mathbf{q'}_1\mathbf{q'}_2}$ but not in $\overline{\mathbf{q''}_1\mathbf{q''}_2}$.

According to the above definition, to obtain a closed form solution for the distance between a point $\mathbf{p}$ and a line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$, we note that, if the distance is d, then the point lies on the line segment whose end-points are $\mathbf{q}_1 = \mathbf{p}_1 + d \cdot \mathbf{n}_1$ and $\mathbf{q}_2 = \mathbf{p}_2 + d \cdot \mathbf{n}_2$, as illustrated in Figure 10. This line segment is an *iso-segment* whose points are all located at a distance d from the generator line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$. The distance is found as the distance for which the area of the triangle $\mathbf{q}_1\mathbf{q}_2\mathbf{p}$ is zero, i.e. that the cross-product of $\mathbf{q}_1 - \mathbf{p}$ and $\mathbf{q}_2 - \mathbf{p}$ is zero. This leads to the following equation:

$$(\mathbf{p}_1 + d \cdot \mathbf{n}_1 - \mathbf{p}) \times (\mathbf{p}_2 + d \cdot \mathbf{n}_2 - \mathbf{p}) = 0, \tag{23}$$

that reduces to the form

$$\mathbf{a} + d \cdot \mathbf{b} + d^2 \cdot \mathbf{c} = 0. \tag{24}$$

Equation 23 is a system of three quadratic equations with a single unknown d. Any of these equations can be used to compute d after making sure that the chosen parameters do not vanish altogether. The chosen equation can have up to two real roots. If the number of real roots is zero, the point $\mathbf{p}$ is located outside the cell. If there is a single real root, the point is located inside the cell and the distance is valid. If there are two real roots, the root of interest is the one with the smallest absolute value. The second root is related to the phenomenon of fundamental cell self-intersection (see Figures 11 and 14) which is explained later in this section. At this point it is sufficient to mention that the other root is larger than the maximal allowed distance within the cell. The validity of the root can also be verified by testing whether the point $\mathbf{p}$ is contained in line segment $\overline{\mathbf{q}_1\mathbf{q}_2}$ or not (see Figure 11).

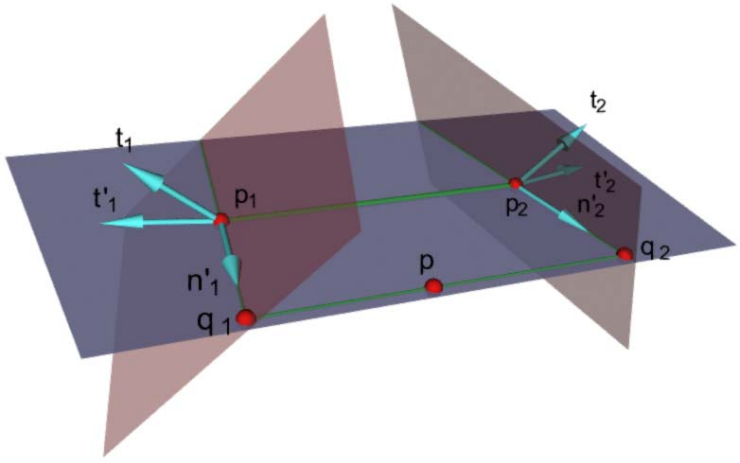


Fig. 12. Computing the distance in the direction of interpolated normals for non-planar curves. Before computing the distance, the tangents $\mathbf{t}_1$ and $\mathbf{t}_2$ are projected onto the plane containing $\mathbf{p}_1, \mathbf{p}_2$ and $\mathbf{p}$. The projections $\mathbf{t}'_1$ and $\mathbf{t}'_2$ are then used in equations 19 through 23 instead of $\mathbf{t}_1$ and $\mathbf{t}_2$.

Extending the above developments to the 3D case is straightforward. It is sufficient to project the tangents and normals on the plane containing $\mathbf{p}_1, \mathbf{p}_2$ and $\mathbf{p}$ as illustrated in Figure 12. Projected tangents $\mathbf{t}'_1$ and $\mathbf{t}'_2$ as well as projected normals $\mathbf{n}'_1$ and $\mathbf{n}'_2$ are then used in equations 19 through 23 instead of $\mathbf{t}_1$ and $\mathbf{t}_2$.

Choosing the definition for the distance allows one to describe the shape of the fundamental cell. To do so, note that the projected normals $\mathbf{n}'_1$ and $\mathbf{n}'_2$ at the two end-points of the line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$ delimit the cell in the plane $\mathbf{p}_1\mathbf{p}_2\mathbf{p}$, (see Figures 10 and 12). Normals projected on all planes containing the line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$ are located in the two planes perpendicular to the two tangents (shaded in red in Figure 12). The cell is therefore delimited by these two planes and by the maximal prescribed distance. In general, the fundamental cells have a cylindrical form. An example of a fundamental cell is shown in Figure 13.

The computation of the distance as defined in Eq. 23 requires projecting normals on the plane $\mathbf{p}_1\mathbf{p}_2\mathbf{p}$. In practice projected normals are computed by noting that they are contained in the plane $\mathbf{p}_1\mathbf{p}_2\mathbf{p}$ as well as in the delimiting planes perpendicular to tangents $\mathbf{t}_1$ and $\mathbf{t}_2$. Therefore, the two normals are given as:

$$\begin{aligned} \mathbf{n}'_1 &= \mathbf{t}_1 \times \mathbf{n}_p, \\ \mathbf{n}'_2 &= \mathbf{t}_2 \times \mathbf{n}_p, \end{aligned} \tag{25}$$

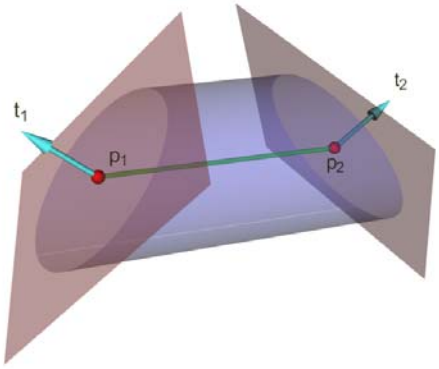
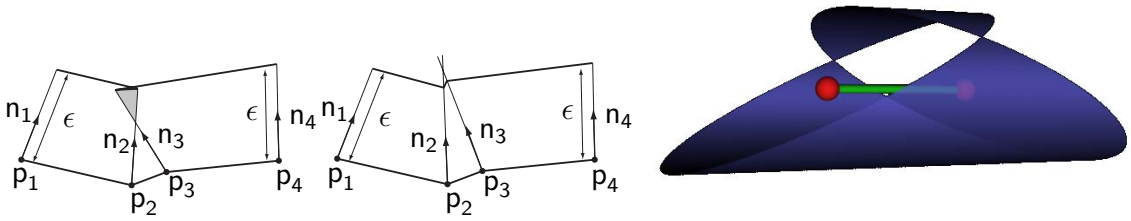


Fig. 13. Example of a fundamental cell for a line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$. The cell is delimited by two planes perpendicular to the tangents $\mathbf{t}_1$ and $\mathbf{t}_2$ at end-points of the line segment. The size of the cylindrical surface delimiting the cell depends on the maximal prescribed distance (envelope size).



a) b) c)

Fig. 14. Self-intersection of a fundamental cell. a) Due to noise, normals $\mathbf{n}_2$ and $\mathbf{n}_3$ meet inside the envelope, making the distance invalid in a part (shaded region) of the cell associated with line segment $\overline{\mathbf{p}_2\mathbf{p}_3}$. b) Filtering the tangents (and thus the normals) makes the intersection appear outside the envelope, thus making the distance valid for all points within the envelope. c) Envelope of a self-intersecting fundamental cell in 3D space.

where $\mathbf{n}_p$ is the normal on the plane $\mathbf{p}_1\mathbf{p}_2\mathbf{p}$.

Noisy data and insufficiently filtered tangents can lead fundamental cells to self-intersect as well as to intersect neighbouring cells; this is illustrated in Figure 14a and c. The distance is not defined for these regions and should not be calculated. There are two ways to circumvent this problem. The first way is to increase the level of filtering of the tangents, thus making the intersections appear outside the envelope, as illustrated in Figure 14b. For the limit case, the tangents become parallel and the cell will never self-intersect. The second way consists in rejecting all voxels that are not located on the right side of the delimiting planes. If the orientation of tangents is assumed to be as shown in Figure 12, a point $\mathbf{p}$ belongs to a valid region of the cell associated with line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$ when the two following conditions are met:

$$\begin{aligned} \langle \mathbf{p}_1 - \mathbf{p}, \mathbf{t}_1 \rangle &\leq 0, \\ \langle \mathbf{p}_2 - \mathbf{p}, -\mathbf{t}_2 \rangle &\leq 0, \end{aligned} \tag{26}$$

since the tangents are normal to delimiting planes.

Equation 21 is valid only inside the fundamental cell of a line segment. Prior to computing the distance to the curve at some point p, one has to find to which cell the point $\mathbf{p}$ belongs. This is a tedious task that can be avoided by inverting the process and by rather finding all voxels falling inside each cell separately. By doing so, the complexity of field computation is linear with respect to the number of line segments (measured points).

Computing the field for a cell reduces therefore to a simple two-step process. First, the bounding box of a cell is computed. The two defining corners of the bounding box can be obtained as minimum and maximum values of $\mathbf{p}_1 + \epsilon, \mathbf{p}_1 - \epsilon, \mathbf{p}_2 + \epsilon, \mathbf{p}_2 - \epsilon$. Then, for all grid points located inside the bounding box, it is verified whether they are located between the two delimiting planes as in Eq. 26. If it is so, the distance is computed and accepted if it is smaller than the maximum allowed distance.

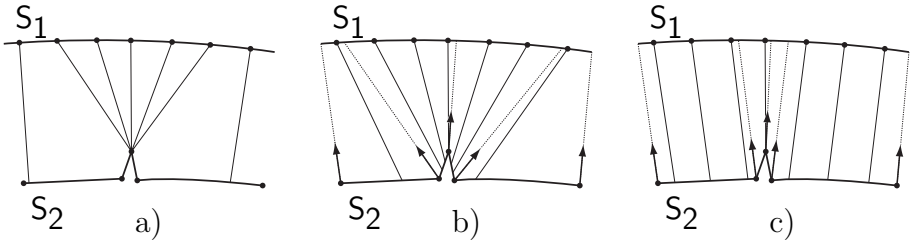


Fig. 15. The effect of noise on the matching step of registration. a) Matching with the noisy curve (or surface) S_2 using Euclidean distance. Since the noisy point in this example is closest to S_1, it attracts most of the points. b) Matching using the distance defined in Eq. 21 with unfiltered normals (depicted as arrows). c) Matching in the direction of filtered normals. Matched points are more evenly distributed over S_2.

The effect of choosing the distance as defined in Eq. 21 on matching in presence of noise, is illustrated in Figure 15.b and c. Filtering the tangents, and at the same time normals, alters the direction towards the closest point, and therefore matching directions. As a consequence, the matched points are distributed more evenly over the curve as well as over the reconstructed surface. It is important to note that only tangents on the curves are filtered, the measured 3D data remaining unchanged.

The procedure for computing the field is further detailed in the pseudo-code below. It is assumed that the normals and tangents are computed prior to field computation.

Data : Set of curves $\alpha_i = \{\mathbf{p}_{i,1}, \ldots \mathbf{p}_{i,N_i}\}$ and tangent $\mathbf{t}_{i,j}$ at each point $\mathbf{p}_{i,j}$
Initialize matrices C at all voxels to zero;
for *i = 1:Number of curves* **do**
  for *j = 1:$N_i - 1$ (number of points in the curve i)* **do**
    Compute the bounding box for the line segment $\overline{\mathbf{p}_{i,j}\mathbf{p}_{i,j+1}}$
    for *all voxels* $\mathbf{p}_v$ *in the bounding box* **do**
      if $\mathbf{p}_v$ *is in the fundamental cell associated with* $\overline{\mathbf{p}_j\mathbf{p}_{j+1}}$, *(Eq. 26)* **then**
        Compute the projected normals $\mathbf{n}'_{i,j}$ and $\mathbf{n}'_{i,j+1}$ (Eq. 25)
        Compute the distance between $\mathbf{p}_v$ and the line segment (Eq. 23)
        Compute the parameter u_c such that Eq. 22 is satisfied
        Compute the tangent $\mathbf{t}(u_c)$ (Eq. 19) at the closest point $\mathbf{p}_c$
        Update matrix C at voxel $\mathbf{v}$ (Eq. 13).
        Update the sum of distance vectors $\tilde{\mathbf{v}}$ with $\mathbf{p}_c - \mathbf{p}_v$ (Eq. 9)
      end
    end
  end
end

Algorithm 1: Computation of the vector field from a set of curves.

3.5 Registration using vector fields

Unlike range images, surface curves cannot be registered with each other one pair at a time. The number of intersections between two curves depends on the shape of the curves and is usually insufficient to compute the rigid transformation needed to align them. In some cases, for instance in registering surface profiles, such an approach would simply make the curves overlap. Furthermore, registering all curves simultaneously has $O(N^2)$ complexity with respect to the number of curves and quickly becomes limiting due to the large number of curves that is needed to reconstruct an object. An elegant and efficient way to circumvent these problems is to register curves to the *reconstructed model*. Since the vector field contains all the information needed for matching, the computational complexity remains linear with respect to the number of curves.

Once the vector field is computed for a reasonable number of curves, registering a curve becomes straightforward: for a control point $\mathbf{p} = [x, y, z]^T$ on a curve, the corresponding point $\mathbf{p}_c$ on the reconstructed surface is given as the value of the vector field at the closest voxel $\mathbf{p_v}$ to the point $\mathbf{p}$:

$$\mathbf{p}_c = \mathbf{p} + \mathbf{f}(\mathbf{p_v}), \tag{27}$$

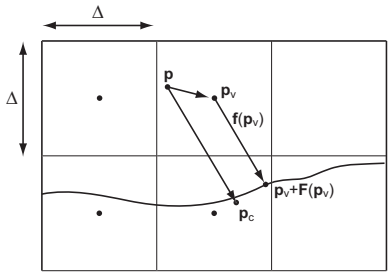


Fig. 16. Matching a point $\mathbf{p}$ using the closest voxel centre $\mathbf{p}_v$. The closest point $\mathbf{p}_c$ is obtained using Eq. 29.

This closest voxel $\mathbf{p}_v$ is computed as

$$\mathbf{p}_v = \lfloor \mathbf{p} \rfloor = [\lfloor x/\Delta + 0.5 \rfloor, \lfloor y/\Delta + 0.5 \rfloor, \lfloor z/\Delta + 0.5 \rfloor]^T, \tag{28}$$

where Δ is the voxel size (see Figure 16). Clearly, the voxel size introduces an error that can be reduced by linearly interpolating the value of the field at the voxel $\mathbf{p}_v$, i.e.

$$\mathbf{p}_c = \mathbf{p} + \mathbf{f}(\mathbf{p_v}) + \frac{\mathbf{f}(\mathbf{p_v}) < \mathbf{f}(\mathbf{p}_v), (\mathbf{p} - \mathbf{p}_v) >}{\|\mathbf{f}(\mathbf{p}_v)\|^2}. \tag{29}$$

This correction is illustrated in Figure 16. Note that since it is a linear interpolation, the closest point will in general not be located exactly on the surface. However, as demonstrated in the next section, the error is small.

The curves can be registered to a completely or partially reconstructed model, thus providing two strategies for registration: simultaneous and incremental. For simultaneous registration, the model is first reconstructed from all available curves. That is, the vector field is computed using all curves. Then, each curve is registered to this model, one at a time. Finally, the vector field is recomputed and the whole procedure is repeated until no further improvement is possible. In the case of incremental registration, an initial model is created from a relatively small number of curves. Then, the next curves are first registered and then integrated into the model.

As explained earlier, the field is updated by updating matrix C (see Eq. 12) at each voxel that is located inside the envelope of at least one curve. Even though the field is represented indirectly through the matrix C, in the pseudo-code below we will refer to it as the vector field $\mathbf{f}$. Two strategies are described in the following pseudo-code.

```
Initialize field f to zero;
for i = 1:M (number of curves used to create an initial model) do
    Compute the field f_i for curve i and add it to f;
end
i ← M + 1;
repeat
    repeat
        Find the matching points for the control points of curve i;
        Compute and apply the rigid transformation on curve i;
    until until convergence;
    Compute the field f_i for curve i and add it to f;
    i ← i + 1;
until no curves left;
```

Algorithm 2: Incremental registration

```
repeat
    Initialize field f to zero;
    for i = 1:Number of curves do
        Compute the field f_i for curve i and add it to f;
    end
    for i = 1:Number of curves do
        repeat
            Find the matching points for the control points of curve i;
            Compute and apply the rigid transformation on curve i;
        until until convergence;
    end
until until convergence;
```

Algorithm 3: Simultaneous Registration

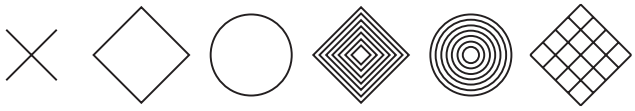


Fig. 17. Light patterns used in experiments.

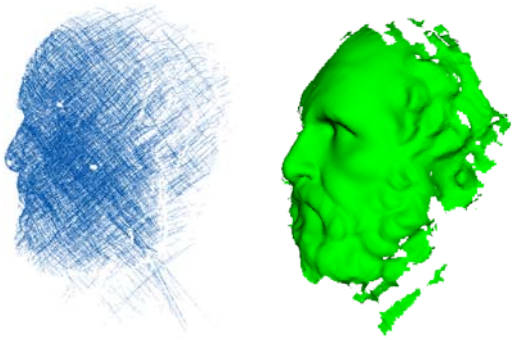


Fig. 18. Reconstruction from real range data. a) Raw range data (cross-hair patterns measured on the surface of the model). b) Reconstructed surface.

4 Results

The quality of the modeling algorithm has been assessed by experimenting under varying levels of noise, registration errors, and different laser patterns using both synthetic and real range data. The results obtained in these experiments are presented in this section.

4.1 Reconstruction experiments

An example of reconstruction from multiple curves, using real range data is shown in Figure 18b. The data for this model consists of 600 curves using the 3D sensor described in [7] which acquires surface curves by measuring distances to a cross-hair laser pattern (Fig. 17a) projected on the surface of an object.

Figure 19 depicts an example of incremental reconstruction using a set of circular light patterns. The figure also illustrates another important aspect of reconstruction: surface filtering by integrating redundant data. When the curves are well registered, adding more curves reduces the variance of noise while preserving fine details. To further illustrate this concept, the reconstruction error has been measured as a function of the number of integrated curves (see Figure 20). As a measure of reconstruction error we adopted the distance of each vertex of the reconstructed surface to the closest point on the reference

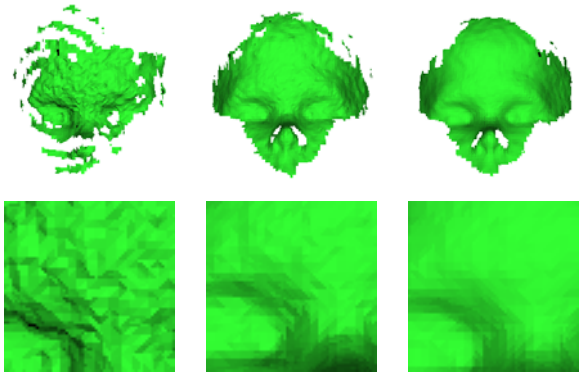


Fig. 19. Filtering by averaging redundant data. From left to right: reconstruction from 10, 120 and 240 curves. Bottom row: left eye detail of the reconstructed model.

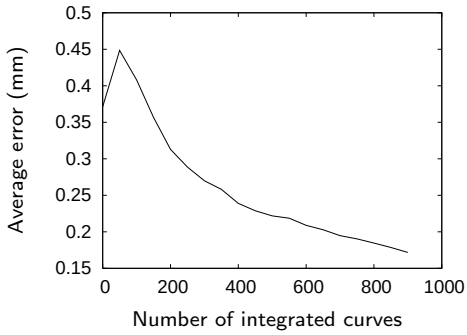
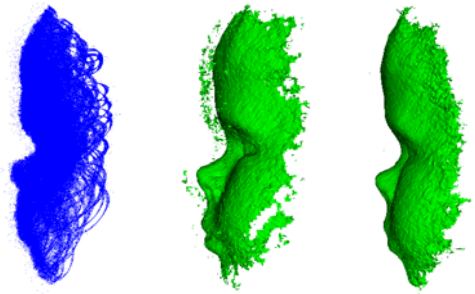


Fig. 20. Average error of the reconstructed surface as a function of the number of curves.

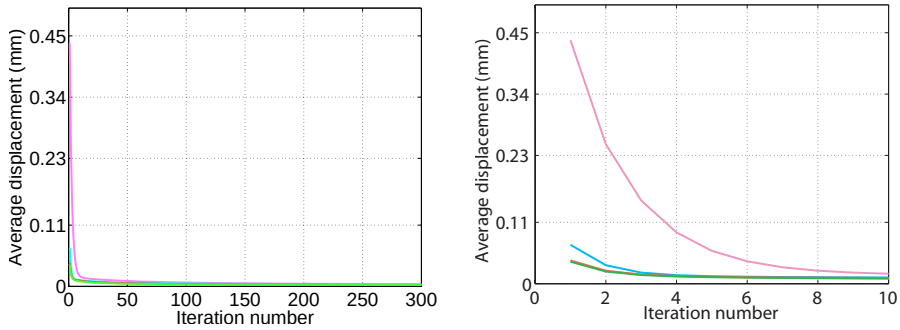
model. The reference model of the head (the same model used to generate synthetic range data) is a highly accurate model measured with a range sensor whose precision is $25\mu m$.

The most important parameter for reconstruction is the size of the envelope (ϵ). On the one hand, since the reconstruction performs averaging in a neighbourhood whose size is equal to ϵ, the envelope should be as small as possible to prevent a loss of fine details. Furthermore, increasing the size of the envelope slows down the algorithm since the number of points at which the field is computed grows as a power of two with the size of the envelope. On the other hand, too small an envelope results in poor performance in the presence of noise and registration errors. In particular, if the envelope is smaller than the level of noise (especially outliers), small isolated patches appear around the reconstructed surface, as illustrated in Figure 21b. A solution to this problem is to increase the size of the envelope (Figure 21c), but with the aforementioned performance penalty and detail loss. Alternatively, the small patches can be removed easily in a post-processing step since they are not connected to the main surface.



(a) (b) (c)

Fig. 21. Influence of high level of noise and outliers on the reconstructed surface (simulation). a) Noisy range data with 1% outliers. b) If the level of noise is larger than the size of the envelope, small isolated patches appear around the surface. c) Increasing the size of the envelope removes the isolated patches.



(a) (b)

Fig. 22. Average displacement of points as a function of the number of iterations. a) Inner loop of the registration algorithm (Algorithm 3). b) Evolution of the inner loop convergence during 5 iterations of the outer loop.

4.2 Registration experiments

As shown in Figure 22, the registration step of the algorithm converges quickly and remains stable over a large number (300) of iterations, (see Figure 22a). Convergence has been tested by measuring the average displacement of circular curves (17c) as a function of the number of iterations. Other patterns behave similarly. From algorithm 3, the inner loop converges in less than 30 iterations on average. To visualize the effect of the field update in the outer loop, the convergence curves in a) are drawn for three iterations of the outer loop. One may observe the decrease in the initial state (Figure 22b).

The quality of registration has been assessed by registering synthetic curves with varying levels of noise, outliers and initial registration errors. For that purpose, 6 noisy synthetic data sets have been generated using the model of a head and for 6 curvilinear patterns. Registration errors were introduced by perturbing the pose rotation angles randomly within $[-\delta_r, \delta_r]$ and the translation

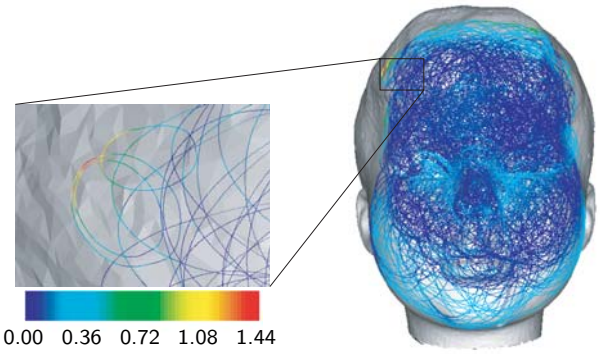


Fig. 23. Colour encoded distribution of residual registration errors (mm). Curves on the border of the scanned area are only partially matched and therefore unreliably registered. Only three curves have a registration error larger than 1mm.

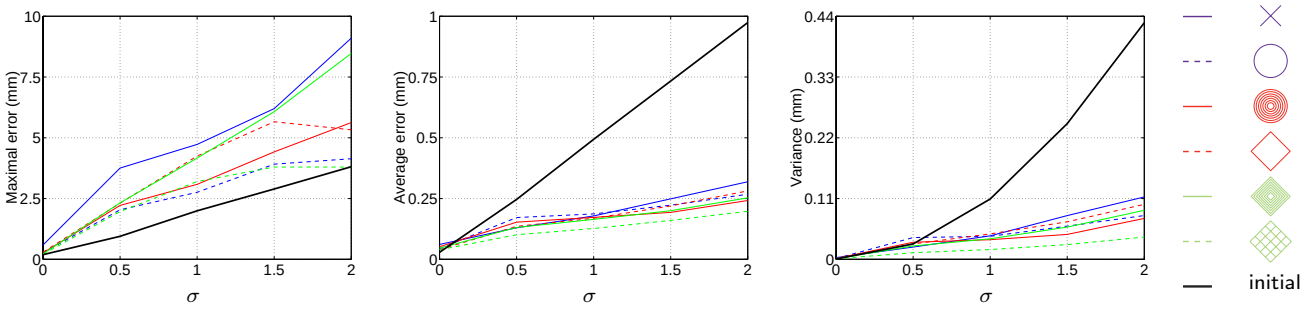
within $[-\delta_t, \delta_t]$. Noise was simulated by translating each point in the direction of the laser projector for a random distance chosen within $[-\delta_n, \delta_n]$ while the outliers were generated by randomly choosing 1% of the total number of points and displacing them for a random distance chosen within $[-\delta_o, \delta_o]$. Random variables follow uniform distributions. The size of the model is approximately 20 cm while the size of a voxel is 1 mm.

The results summarized in Figure 24 show the performance of the registration algorithm as a function of the initial registration errors and the level of noise. Residual registration error has been obtained as the distance between the registered points and their corresponding closest points on the reference model. Prior to computing residual errors, all curves as a whole, i.e. as a rigid set, were registered to the reference model using an ICP algorithm.

The average residual error as well as the variance slowly increase with noise and initial registration errors. An average error of 0.25 mm or below is sufficiently small to yield a reconstructed surface without visible artefacts. More importantly the plots show that the performance of the algorithm gracefully degrades as the level of noise increases rather than to collapse after a certain level of noise.

The large maximum residual error in Figure 24a indicates that some curves were not well registered. On the other hand, small variance and small average error indicate that the number of curves having this high error is small. As illustrated in Figure 23 the curves with large errors are located on the boundary of the scanned area where the density of curves is low. In those regions the field could not be computed so that curves are only partially matched against the reconstructed surface, thus giving rise to an unreliable registration. Among 450 curves, there were only three curves with an error higher than 1mm. This problem can be solved by rejecting curves that are partially matched.

Finally, it should be noted that all laser patterns gave similar results. How-



(a) (b) (c)

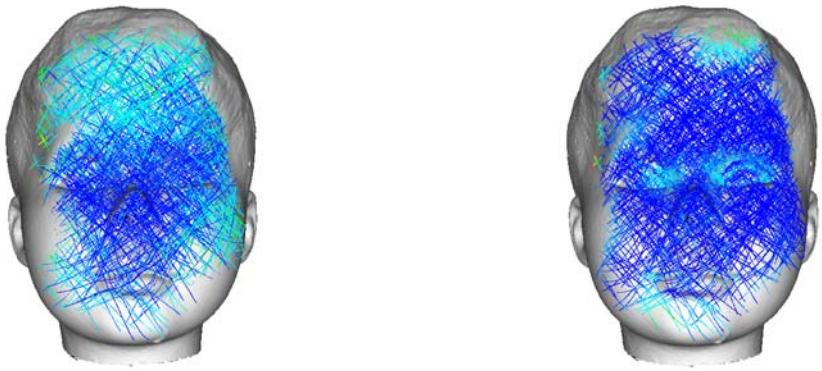
Fig. 24. Residual registration errors as a function of initial displacements of curves and the level of noise. The values δ_r, δ_t, δ_n and δ_o (see the text for their meaning) are related to the X-axis index σ as : $\delta_r = \sigma(deg)$, $\delta_t = \sigma(mm)$, $\delta_n = \sigma(mm)$ and $\delta_o = 5\sigma(mm)$. The thick black line indicates initial registration error.

ever, as shown in Figures 24b and c, patterns having the most uniform and dense distribution of points perform slightly better. A thorough analysis of the different patterns is beyond the scope of this paper.

Another way to evaluate the performance of the registration component of the algorithm is to compute the distance between corrected positions of all points and their exact positions. This is however an incorrect measure since the curves can slide over the surface during registration, thus increasing the distance to the exact positions, but without significantly degrading the final result. For example, a curve measured in a planar region of the surface can be translated and rotated arbitrarily in the same plane without increasing the distance to the surface. This "sliding" effect is illustrated in Figure 25 where the two measures (distance to exact position of each point, and the distance to the closest point on the reference model) are compared. As expected, the distance to the exact positions is higher in relatively flat regions of the surface.

Registering real range data of the head model obtained with the sensor described in [7] data leads to similar results. Initial average registration errors were 0.35 mm with a variance of 0.09 mm^2. After registration, those errors were reduced to 0.26 mm (average) and 0.07 mm^2 (variance), which is in conformity with results obtained with synthetic data. The residual registration error was evaluated using the reference model.

The last experiment shows the performance of the implemented algorithm. The incremental reconstruction from curves having 250 points each is performed in real-time at the frame rate of the sensor (30fps). Thereafter, the registration algorithm is run. Execution time of the inner loop of the registration algorithm 3 for 450 curves is less than 2 seconds in average . Field recomputation takes more time depending on the size of the envelope: 9 sec for $\epsilon = 3$ mm and 23 sec for $\epsilon = 5$ mm. The total registration and reconstruction time is 55 sec for $\epsilon = 3$ mm and 140 sec for $\epsilon = 5$ mm. In all cases the volume grid size was



(a) (b)

Fig. 25. Measuring the residual registration errors. a) The residual registration error (colour encoded, brighter colour indicates larger error) measured as the distance between the corrected positions of points and their exact position. The regions of larger errors are featureless regions of the surface, forehead and cheeks. b) The residual registration error measured as the distance towards the closest point on the reference surface. The errors are uniformly distributed over the surface. Large errors are concentrated in low measurement density regions.

$200 \times 200 \times 200$. The execution times were obtained using a PC with 1.2GHz AMD Athlon processor.

5 Conclusion

A volumetric approach for modeling from curves measured on the surface of a free-form object is proposed. Algorithms for both registration and reconstruction are of linear complexity with respect to the number of curves. The reconstruction is incremental thus allowing integration of a single curve at a time. In addition, the reconstruction is order independent. No intermediate surface representation is required: an implicit volumetric representation of the surface is created directly from the curves. Any curvilinear light pattern can be used for this purpose as long as the resulting curves intersect to provide redundant data for registration and reconstruction. Experimental results on both real and synthetic data provide evidence that the algorithm is robust with respect to the noise and registration errors.

Several extensions with respect to efficiency and generality of the presented algorithms and vector field object representation can be suggested. First, the surface reconstruction algorithm from curves, along with already proposed algorithms for modeling from unorganized sets of points and range images, allows the reconstruction from range data regardless of its dimensionality. However, volumetric representations of different range data types are not compat-

ible; this means that the created volumetric representation, say, from range images cannot be updated with a set of curves or a set of points and vice versa. Unifying the volumetric representation for all types of range data would greatly simplify the modeling process since it would allow mixing of range data acquired with different range sensors. For example, an object could be scanned using a range image sensor that provides 3D images while filling the hard to reach areas with a hand-held sensor that acquires surface curves.

Another important aspect of a modeling algorithm is the efficient use of computer resources. With respect to computational complexity, the presented algorithm is very efficient, having linear complexity, but the price to pay is a very inefficient use of computer memory. Even though already proposed compression schemes such as run length encoding or hash tables can be used to reduce memory requirements, it is still not as efficient as surface based representations since these methods encode only occupancy of voxels regardless of the geometry of the object. We do believe that compression of volumetric data based on the geometry of the object is possible and it will be, along with the aforementioned problem of unified representation, the object of our future work.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Point set surfaces. *IEEE Visualization 2001*, pages 21–28, October 2001.

[2] P. Besl and N. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February 1992.

[3] S. J. Cunnington and A. J. Stoddart. Self-calibrating surface reconstruction for the modelmaker. In *BMVC'98 proceedings*, volume 2, pages 790–799, 1998.

[4] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH '96 Conference Proceedings*, pages 303–312, August 1996.

[5] M. P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976. 503 pages.

[6] O. Hall-Holt and S. Rusinkiewicz. Stripe boundary codes for real-time structured-light range scanning of moving objects. In *Proceedings of ICCV 2001*, pages II: 359–366, 2001.

[7] P. Hébert. A self-referenced hand-held range sensor. In *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 5–11, May 2001.

[8] P. Hébert and M. Rioux. A self-referenced hand-held range sensor. In *Proceedings of SPIE: Three-Dimensional Image Capture and Applications, The International Society for Optical Engineering*, volume 3313, pages 2–13, January 1998.

[9] A. Hilton and J. Illingworth. Geometric fusion for a hand-held 3d sensor. *Machine vision and applications*, 12:44–51, 2000.

[10] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 71–78, July 1992.

[11] J. Kofman and G. K. Knopf. Registration and integration of narrow and spatiotemporally-dense range views. In *Proceedings of SPIE, Vision Geometry VII*, volume 3454, pages 99–109, July 1998.

[12] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH '87 Conference Proceedings*, 21(4):163–169, 1987.

[13] T. Masuda. Object shape modeling from multiple range images by matching signed distance fields. In *Proceedings of First International Symposium on 3D data Processing Visualization and Transmission (3DPVT)*, volume 1, pages 439–448, June 2002.

[14] M. Proesmans, L. Van Gool, and F. Defoort. Reading between the lines - a method for extracting dynamic 3d with texture. In *Proceedings of ICCV 1998*, pages 1081–1086, 1998.

[15] G. Roth and E. Wibowo. An efficient volumetric method for building closed triangular meshes from 3-d image and point data. In W. Davis, M. Mantei, and V. Klassen, editors, *Graphics Interface*, pages 173–180, May 1997.

[16] M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):344–358, 1995.

[17] D. Tubić, P. Hébert, and D. Laurendeau. A volumetric approach for interactive 3d modeling. In *Proceedings of 3DPVT*, volume 1, pages 150–158, June 2002.

[18] D. Tubić, P. Hébert, and D. Laurendeau. A volumetric approach for interactive 3d modeling. *Computer Vision and Image Understanding*, 92:56–77, 2003.

[19] G. Turk and M. Levoy. Zippered polygon meshes from range images. *SIGGRAPH '94 Conference Proceedings*, 26:311–318, 1994.